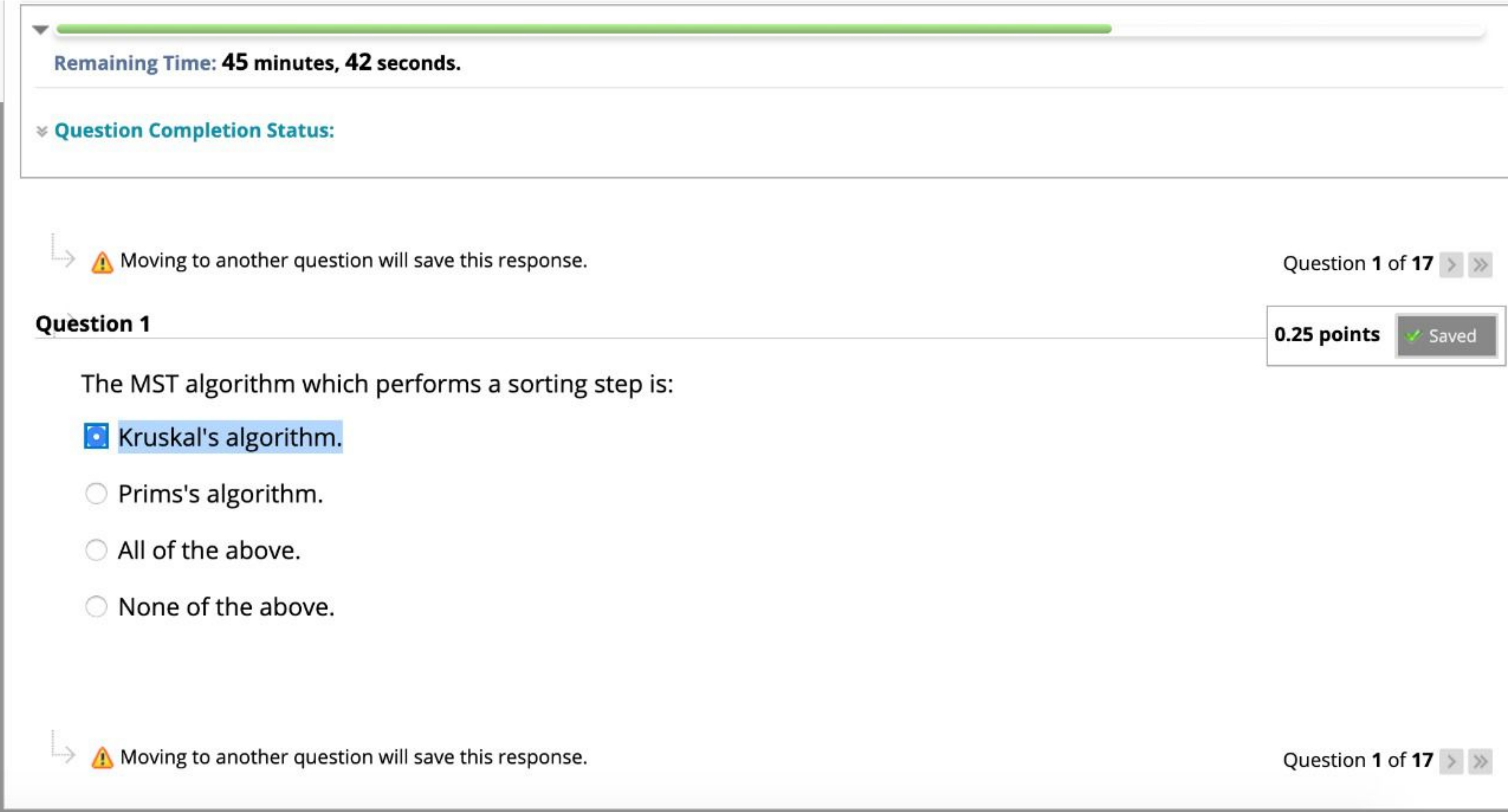


Remaining Time: **45 minutes, 42 seconds.**

Question Completion Status:

⚠ Moving to another question will save this response.

Question **1** of **17**

Question 1

0.25 points ✓ Saved

The MST algorithm which performs a sorting step is:

● Kruskal's algorithm.

○ Prims's algorithm.

○ All of the above.

○ None of the above.

⚠ Moving to another question will save this response.

Question **1** of **17**

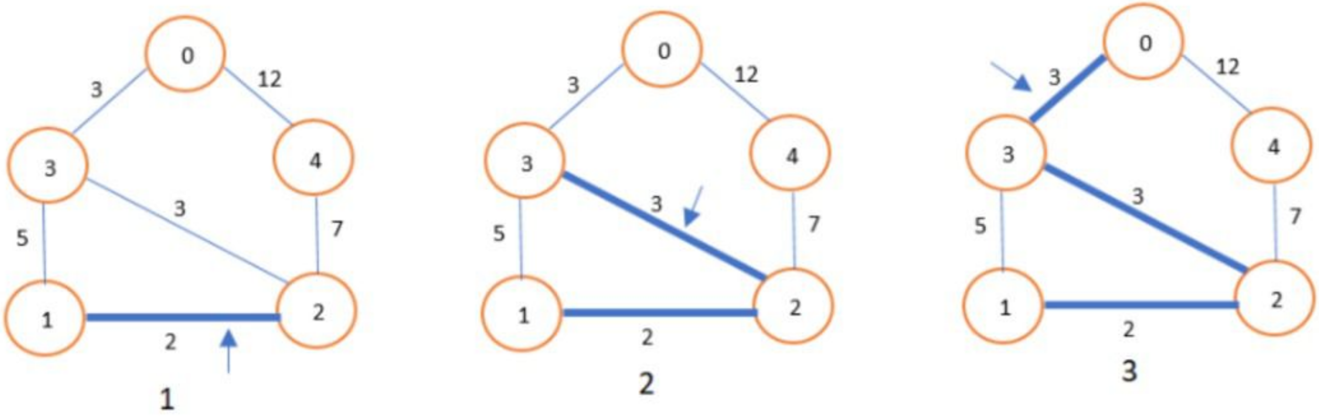
Remaining Time: **42 minutes, 45 seconds.**

Question Completion Status:

We apply Kruskal's algorithm to construct a MST. Follow the steps of the algorithm by filling in the blanks:

a) We maintain a collection VS of disjoint sets of vertices: **VS = {{0},{1},{2},{3},{4}}**

b) we pick edge (1,2), then we pick edge (3,2), and then we pick (3,0) as shown in the following diagram:



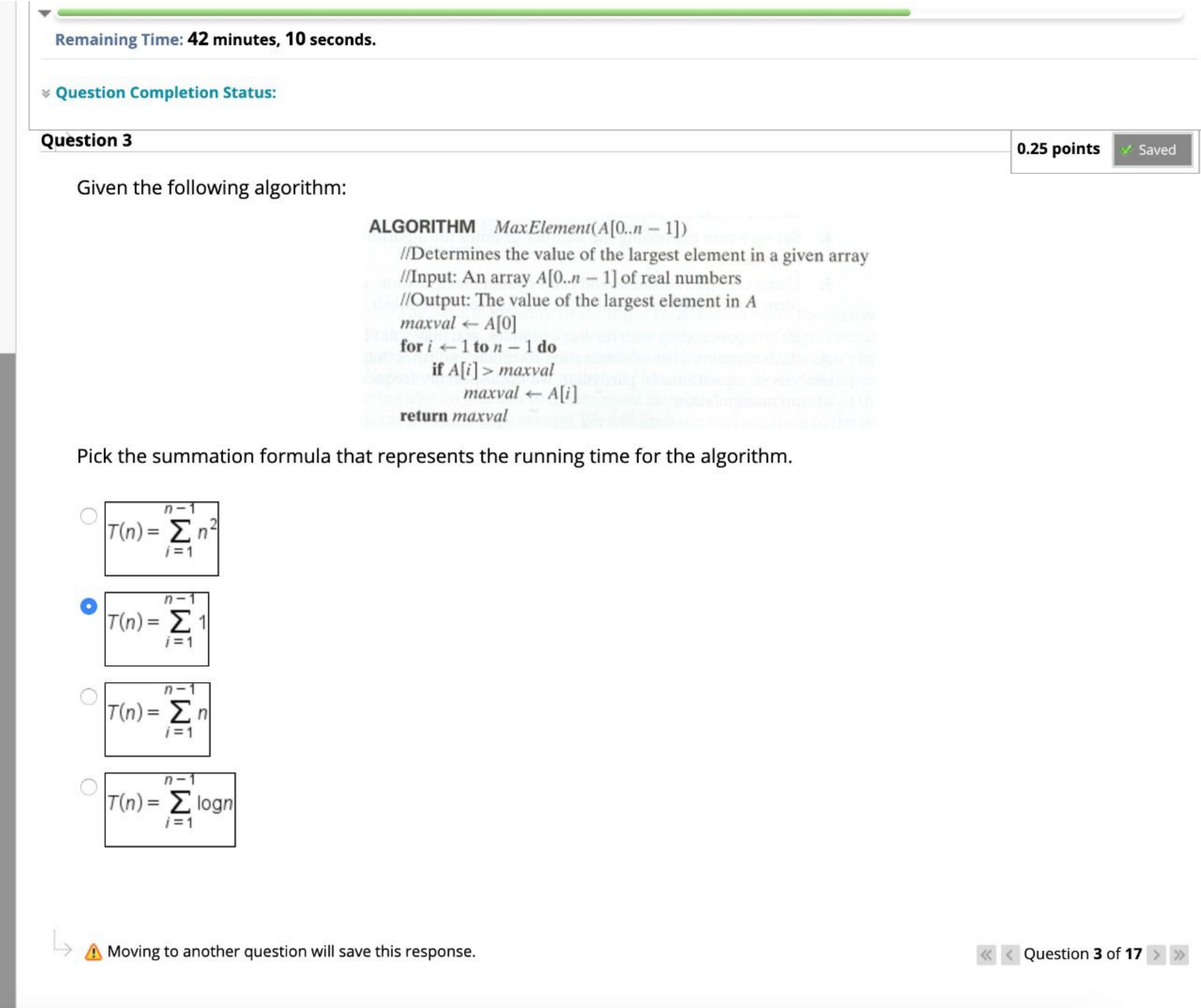
After these steps, the disjoint sets collection becomes: **VS =** **{{0,1,2,3},{4}}** .

c) The algorithm then picks edge (1,3) , but the disjoint set **VS** remains unchanged and therefore, <u>this edge is not added to the MST</u>.

d) The algorithm then picks edge (2,4) and the the dijoint set becomes **VS =** **{{0,1,2,3,4}}** , the algorithm then stops since all the vertices are now in the same set. The resultant MST will have a weight equal to 15 .

⚠ Moving to another question will save this response.

« ‹ Question **2** of **17** › »



Remaining Time: **42 minutes, 10 seconds.**

Question Completion Status:

Question 3

0.25 points ✓ Saved

Given the following algorithm:

```
ALGORITHM  $MaxElement(A[0..n-1])$
    //Determines the value of the largest element in a given array
    //Input: An array $A[0..n-1]$ of real numbers
    //Output: The value of the largest element in $A$
    $maxval \leftarrow A[0]$
    for $i \leftarrow 1$ to $n-1$ do
        if $A[i] > maxval$
            $maxval \leftarrow A[i]$
    return $maxval$
```

Pick the summation formula that represents the running time for the algorithm.

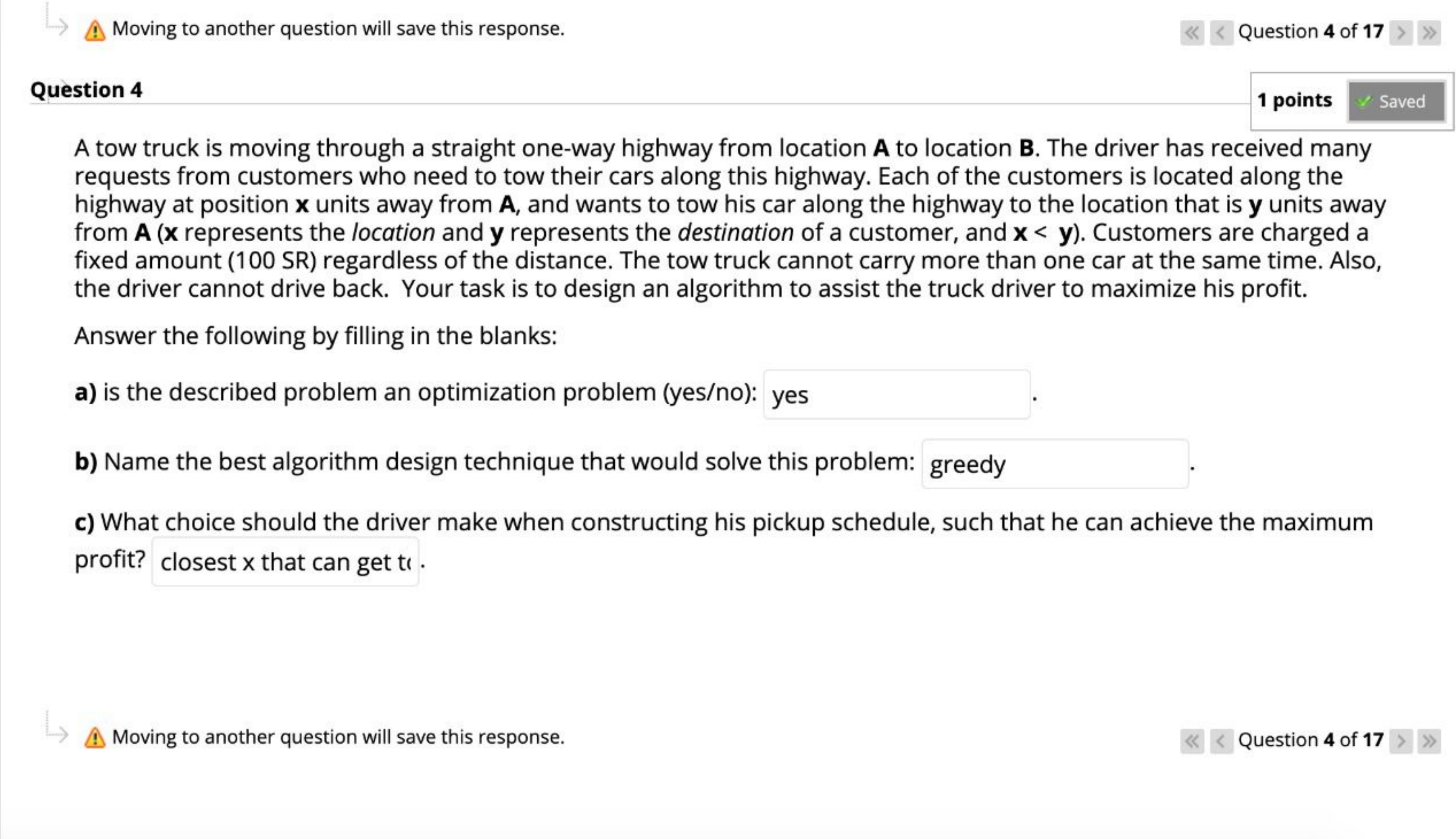
○ $T(n) = \sum_{i=1}^{n-1} n^2$

● $T(n) = \sum_{i=1}^{n-1} 1$

○ $T(n) = \sum_{i=1}^{n-1} n$

○ $T(n) = \sum_{i=1}^{n-1} \text{log}n$

⚠ Moving to another question will save this response.



Question 4

1 points Saved

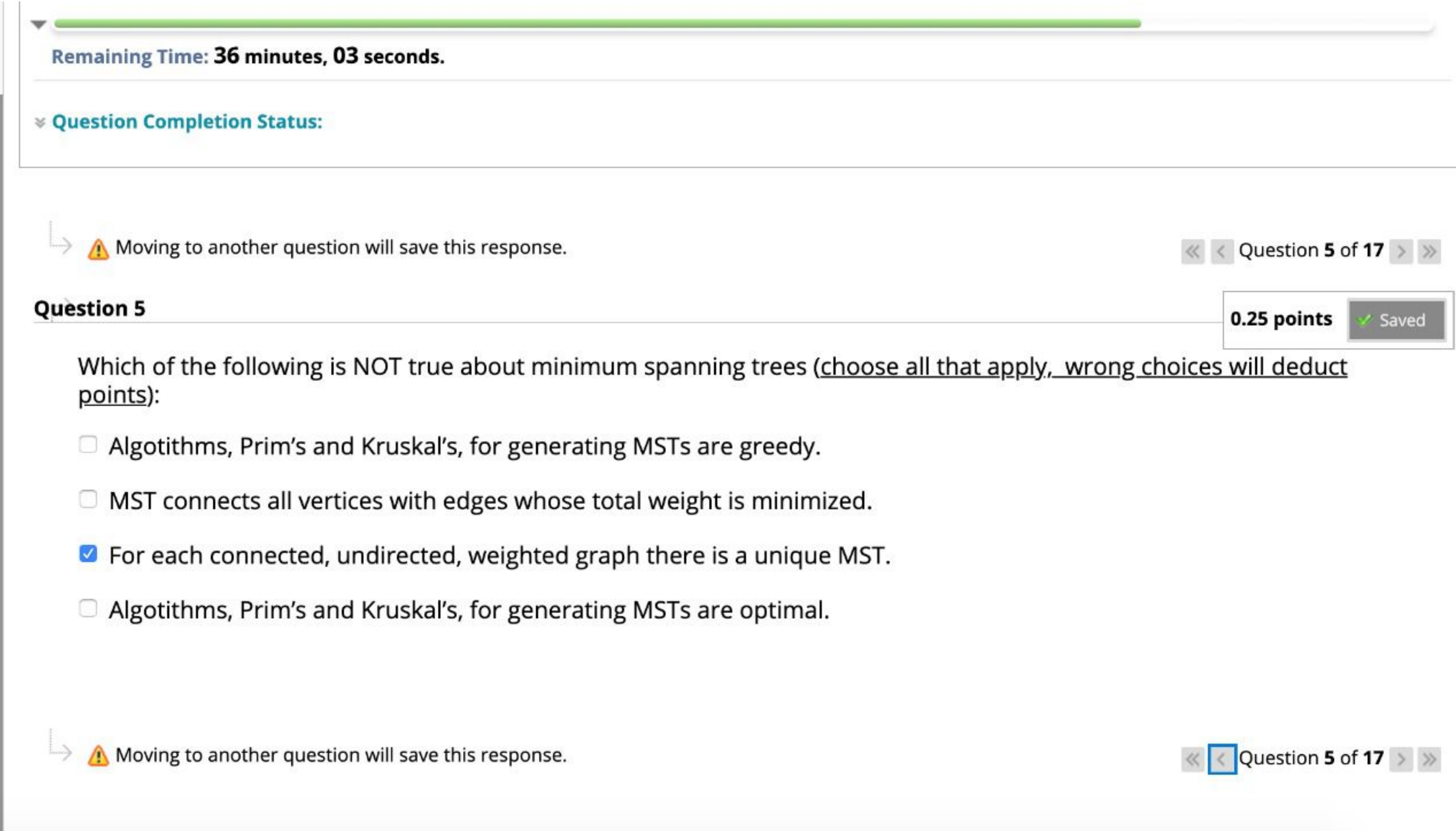
A tow truck is moving through a straight one-way highway from location **A** to location **B**. The driver has received many requests from customers who need to tow their cars along this highway. Each of the customers is located along the highway at position **x** units away from **A**, and wants to tow his car along the highway to the location that is **y** units away from **A** (**x** represents the *location* and **y** represents the *destination* of a customer, and **x** < **y**). Customers are charged a fixed amount (100 SR) regardless of the distance. The tow truck cannot carry more than one car at the same time. Also, the driver cannot drive back. Your task is to design an algorithm to assist the truck driver to maximize his profit.

Answer the following by filling in the blanks:

a) is the described problem an optimization problem (yes/no): yes .

b) Name the best algorithm design technique that would solve this problem: greedy .

c) What choice should the driver make when constructing his pickup schedule, such that he can achieve the maximum profit? closest x that can get t .



Remaining Time: **36 minutes, 03 seconds.**

Question Completion Status:

⚠ Moving to another question will save this response.

Question **5** of **17**

Question 5

0.25 points ✓ Saved

Which of the following is NOT true about minimum spanning trees (choose all that apply, wrong choices will deduct points):

☐ Algotithms, Prim's and Kruskal's, for generating MSTs are greedy.

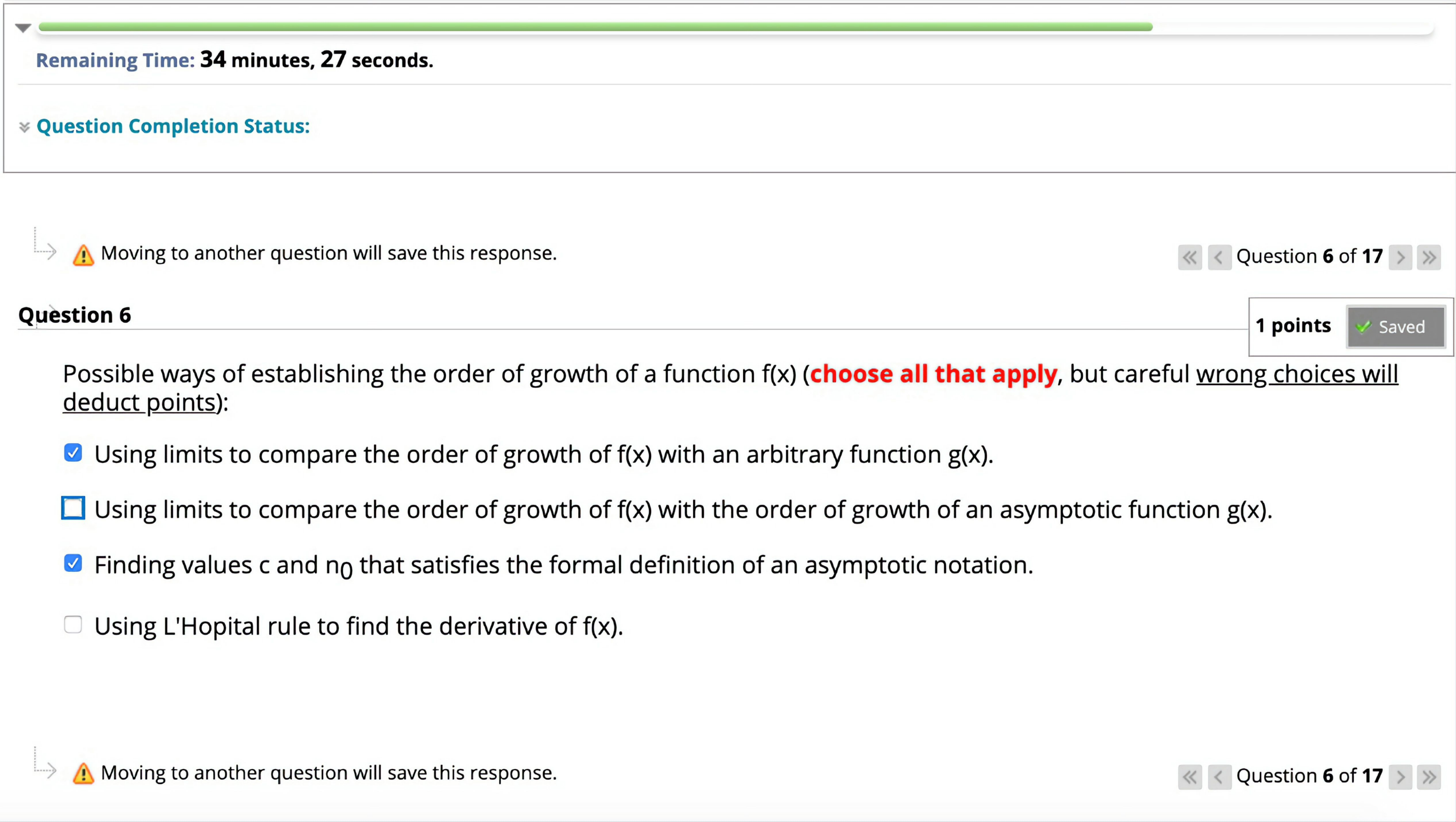
☐ MST connects all vertices with edges whose total weight is minimized.

☑ For each connected, undirected, weighted graph there is a unique MST.

☐ Algotithms, Prim's and Kruskal's, for generating MSTs are optimal.

⚠ Moving to another question will save this response.

Question **5** of **17**



Remaining Time: **34 minutes, 27 seconds.**

Question Completion Status:

⚠ Moving to another question will save this response.

Question **6** of **17**

Question 6

1 points ✓ Saved

Possible ways of establishing the order of growth of a function f(x) (**choose all that apply**, but careful wrong choices will deduct points):

☑ Using limits to compare the order of growth of f(x) with an arbitrary function g(x).

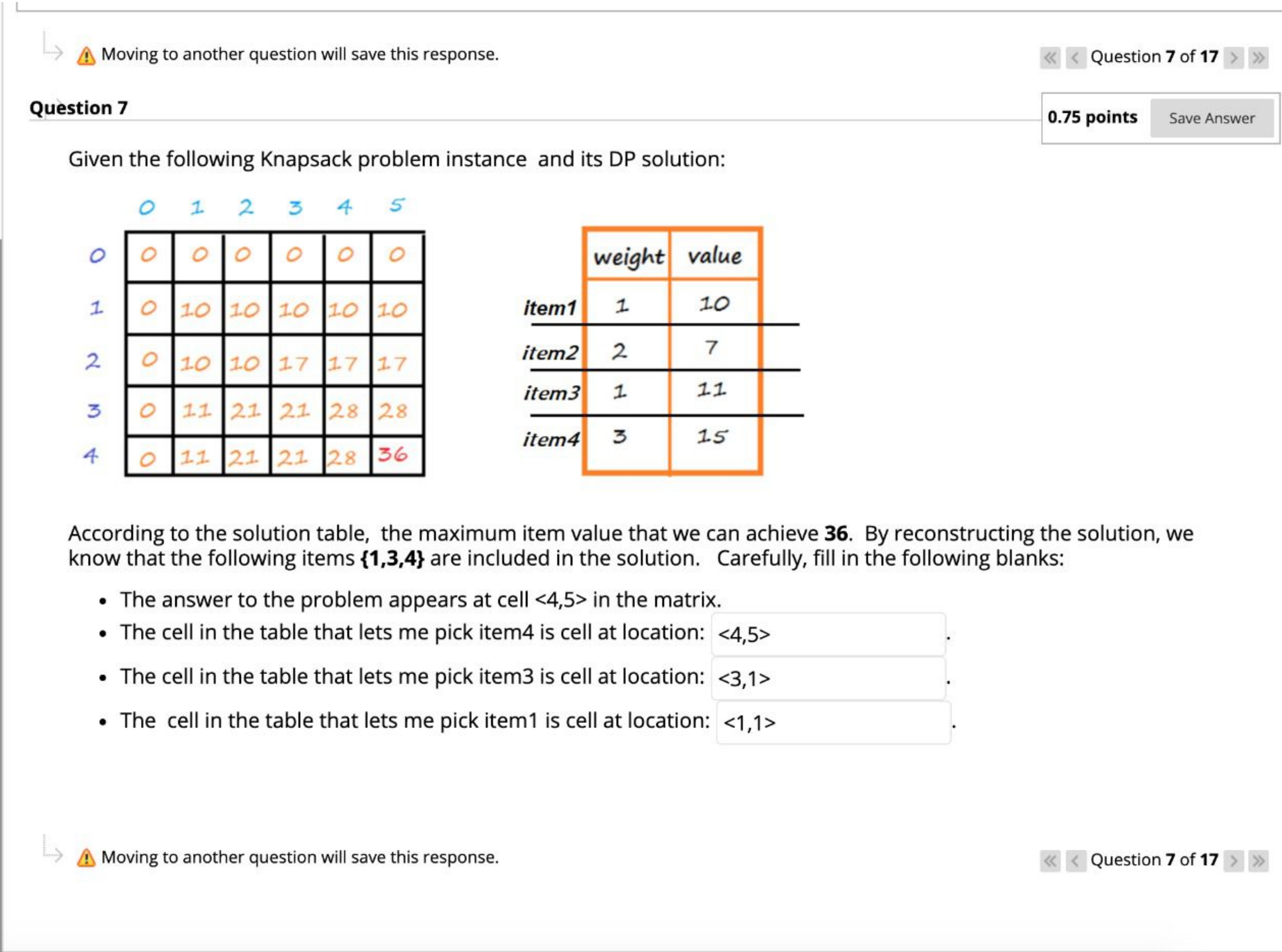
□ Using limits to compare the order of growth of f(x) with the order of growth of an asymptotic function g(x).

☑ Finding values c and n_0 that satisfies the formal definition of an asymptotic notation.

□ Using L'Hopital rule to find the derivative of f(x).

⚠ Moving to another question will save this response.

Question **6** of **17**



Question 7

0.75 points Save Answer

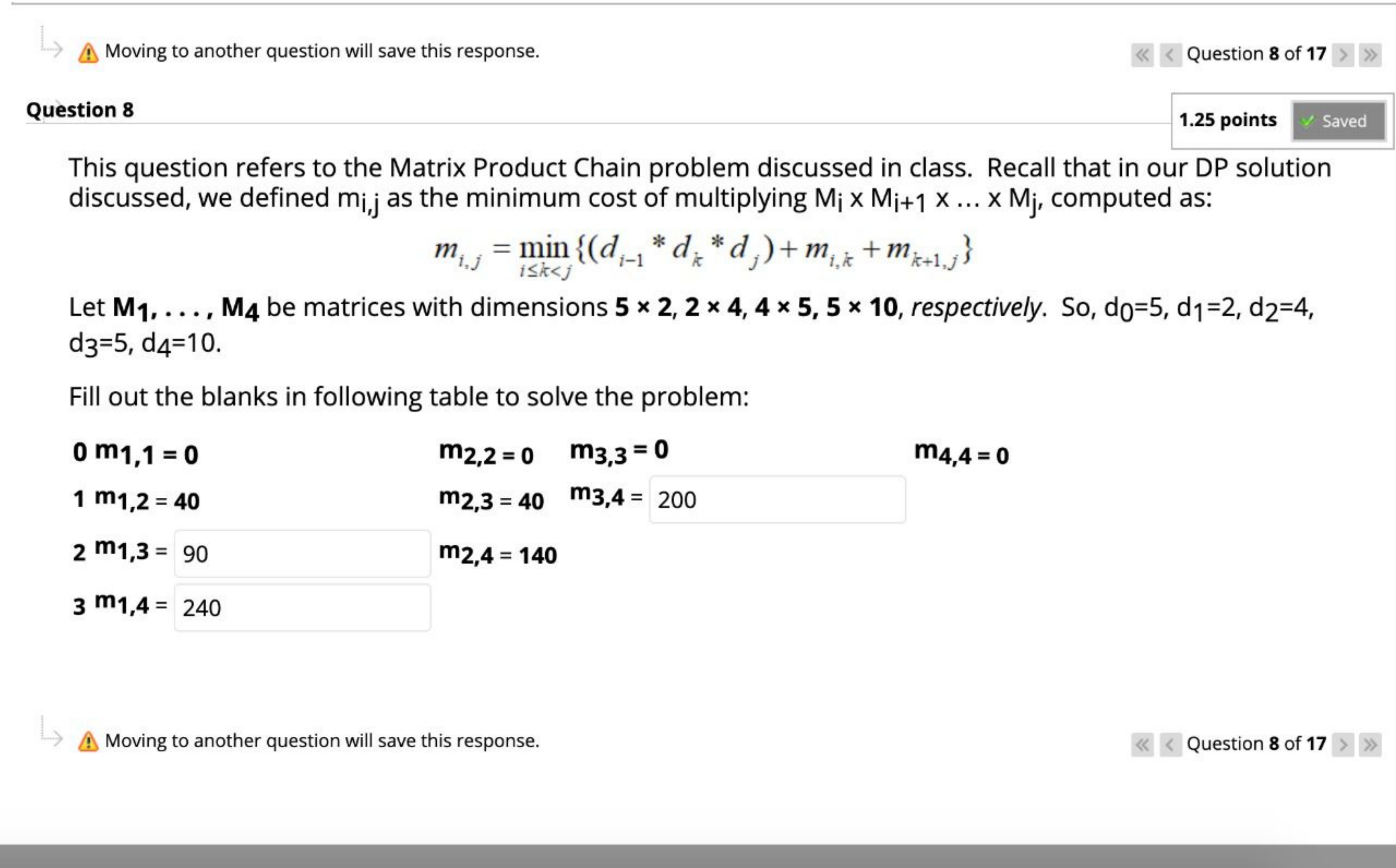
Given the following Knapsack problem instance and its DP solution:

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	10	10	10	10	10
2	0	10	10	17	17	17
3	0	11	21	21	28	28
4	0	11	21	21	28	36

	weight	value
item1	1	10
item2	2	7
item3	1	11
item4	3	15

According to the solution table, the maximum item value that we can achieve **36**. By reconstructing the solution, we know that the following items **{1,3,4}** are included in the solution.   Carefully, fill in the following blanks:

- The answer to the problem appears at cell <4,5> in the matrix.
- The cell in the table that lets me pick item4 is cell at location: <4,5> .
- The cell in the table that lets me pick item3 is cell at location: <3,1> .
- The cell in the table that lets me pick item1 is cell at location: <1,1> .



⚠ Moving to another question will save this response.

« ‹ Question **8** of **17** › »

Question 8

1.25 points ✓ Saved

This question refers to the Matrix Product Chain problem discussed in class. Recall that in our DP solution discussed, we defined $m_{i,j}$ as the minimum cost of multiplying M_i x M_{i+1} x … x M_j, computed as:

$$m_{i,j} = \min_{i \le k < j} \{(d_{i-1} * d_k * d_j) + m_{i,k} + m_{k+1,j}\}$$

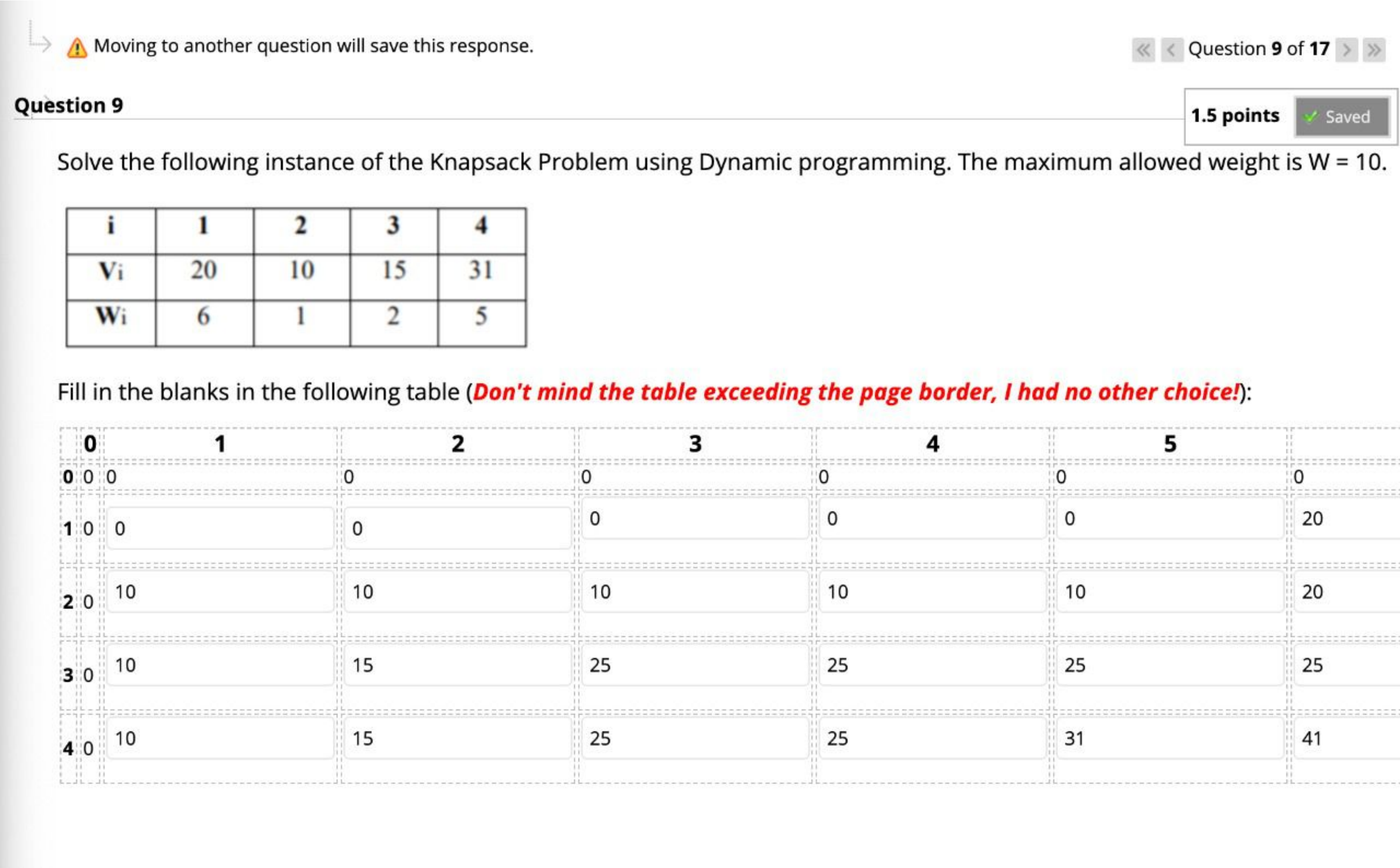
Let $\mathbf{M_1, \ldots, M_4}$ be matrices with dimensions **5 × 2**, **2 × 4**, **4 × 5, 5 × 10**, *respectively*. So, d_0=5, d_1=2, d_2=4, d_3=5, d_4=10.

Fill out the blanks in following table to solve the problem:

0	$\mathbf{m_{1,1} = 0}$	$\mathbf{m_{2,2} = 0}$	$\mathbf{m_{3,3} = 0}$	$\mathbf{m_{4,4} = 0}$
1	$\mathbf{m_{1,2}}$ = **40**	$\mathbf{m_{2,3}}$ = **40**	$\mathbf{m_{3,4}}$ = 200	
2	$\mathbf{m_{1,3}}$ = 90	$\mathbf{m_{2,4}}$ = **140**		
3	$\mathbf{m_{1,4}}$ = 240			

⚠ Moving to another question will save this response.

« ‹ Question **8** of **17** › »



Question 9

1.5 points

Solve the following instance of the Knapsack Problem using Dynamic programming. The maximum allowed weight is W = 10.

i	**1**	**2**	**3**	**4**
Vi	20	10	15	31
Wi	6	1	2	5

Fill in the blanks in the following table (***Don't mind the table exceeding the page border, I had no other choice!***):

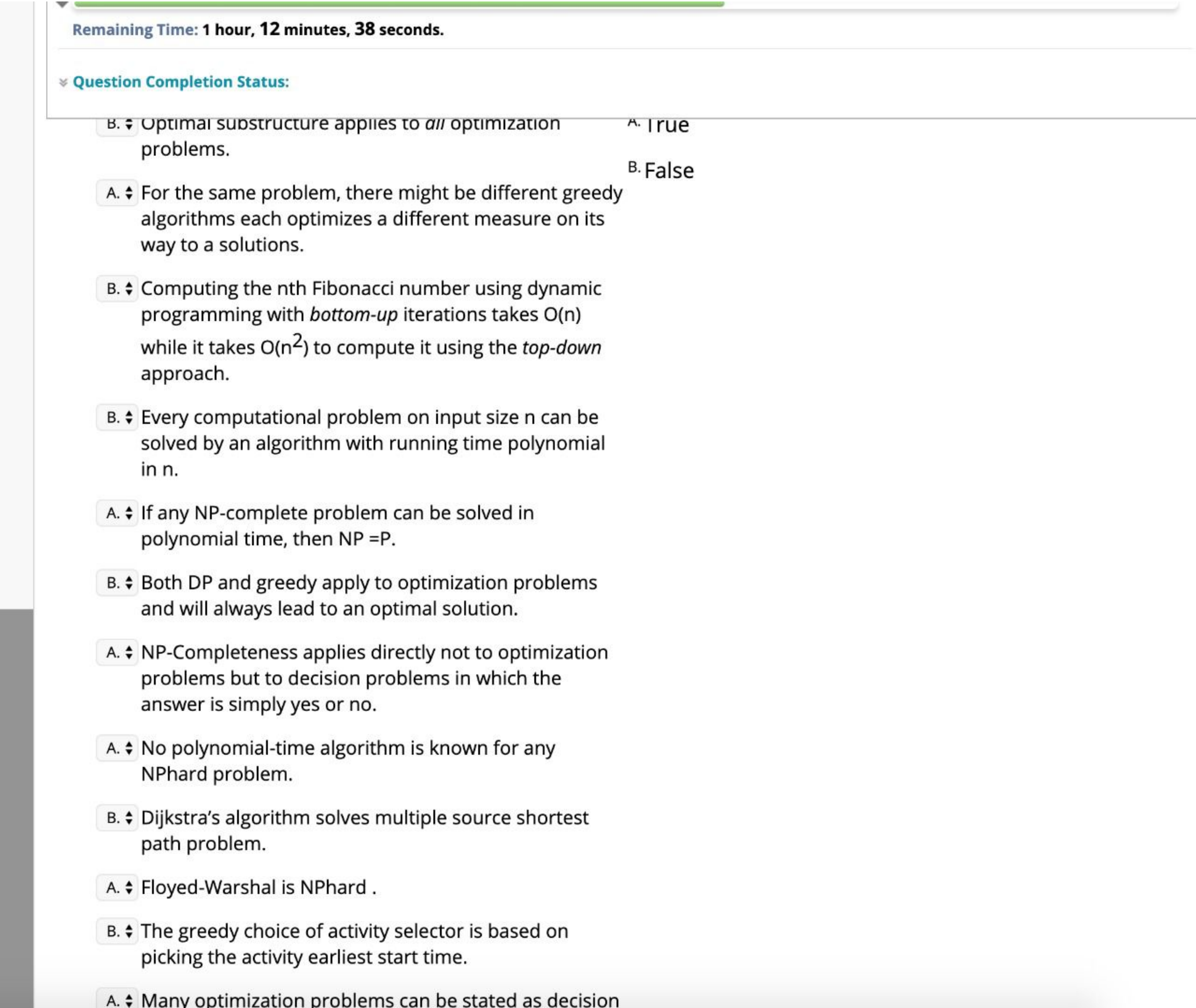
	0	**1**	**2**	**3**	**4**	**5**	
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	20
2	0	10	10	10	10	10	20
3	0	10	15	25	25	25	25
4	0	10	15	25	25	31	41

aximum allowed weight is W = 10.

had no other choice!):

	5	6	7	8	9	10
	0	0	0	0	0	0
	0	20	20	20	20	20
	10	20	30	30	30	30
	25	25	30	35	45	45
	31	41	46	56	56	56

« < Question **9** of **17** > »



Remaining Time: **1 hour, 12 minutes, 38 seconds.**

Question Completion Status:

B. ⬍ Optimal substructure applies to *all* optimization problems.

A. True

B. False

A. ⬍ For the same problem, there might be different greedy algorithms each optimizes a different measure on its way to a solutions.

B. ⬍ Computing the nth Fibonacci number using dynamic programming with *bottom-up* iterations takes O(n) while it takes O(n^2) to compute it using the *top-down* approach.

B. ⬍ Every computational problem on input size n can be solved by an algorithm with running time polynomial in n.

A. ⬍ If any NP-complete problem can be solved in polynomial time, then NP =P.

B. ⬍ Both DP and greedy apply to optimization problems and will always lead to an optimal solution.

A. ⬍ NP-Completeness applies directly not to optimization problems but to decision problems in which the answer is simply yes or no.

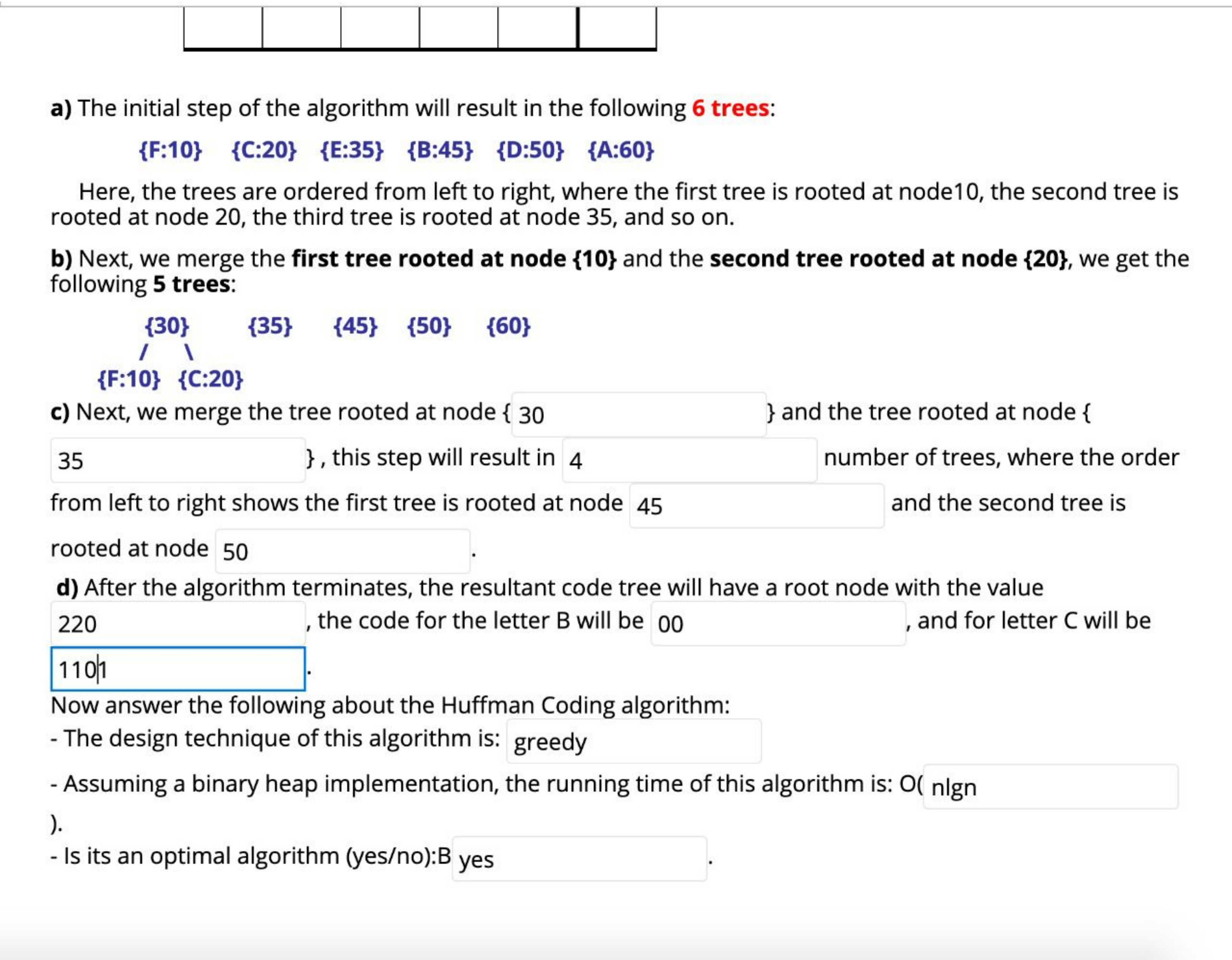
A. ⬍ No polynomial-time algorithm is known for any NPhard problem.

B. ⬍ Dijkstra's algorithm solves multiple source shortest path problem.

A. ⬍ Floyed-Warshal is NPhard .

B. ⬍ The greedy choice of activity selector is based on picking the activity earliest start time.

A. ⬍ Many optimization problems can be stated as decision



a) The initial step of the algorithm will result in the following **6 trees**:

{F:10} {C:20} {E:35} {B:45} {D:50} {A:60}

Here, the trees are ordered from left to right, where the first tree is rooted at node10, the second tree is rooted at node 20, the third tree is rooted at node 35, and so on.

b) Next, we merge the **first tree rooted at node {10}** and the **second tree rooted at node {20}**, we get the following **5 trees**:

```
     {30}       {35}    {45}   {50}    {60}
     /  \
{F:10} {C:20}
```

c) Next, we merge the tree rooted at node { 30 } and the tree rooted at node { 35 }, this step will result in 4 number of trees, where the order from left to right shows the first tree is rooted at node 45 and the second tree is rooted at node 50 .

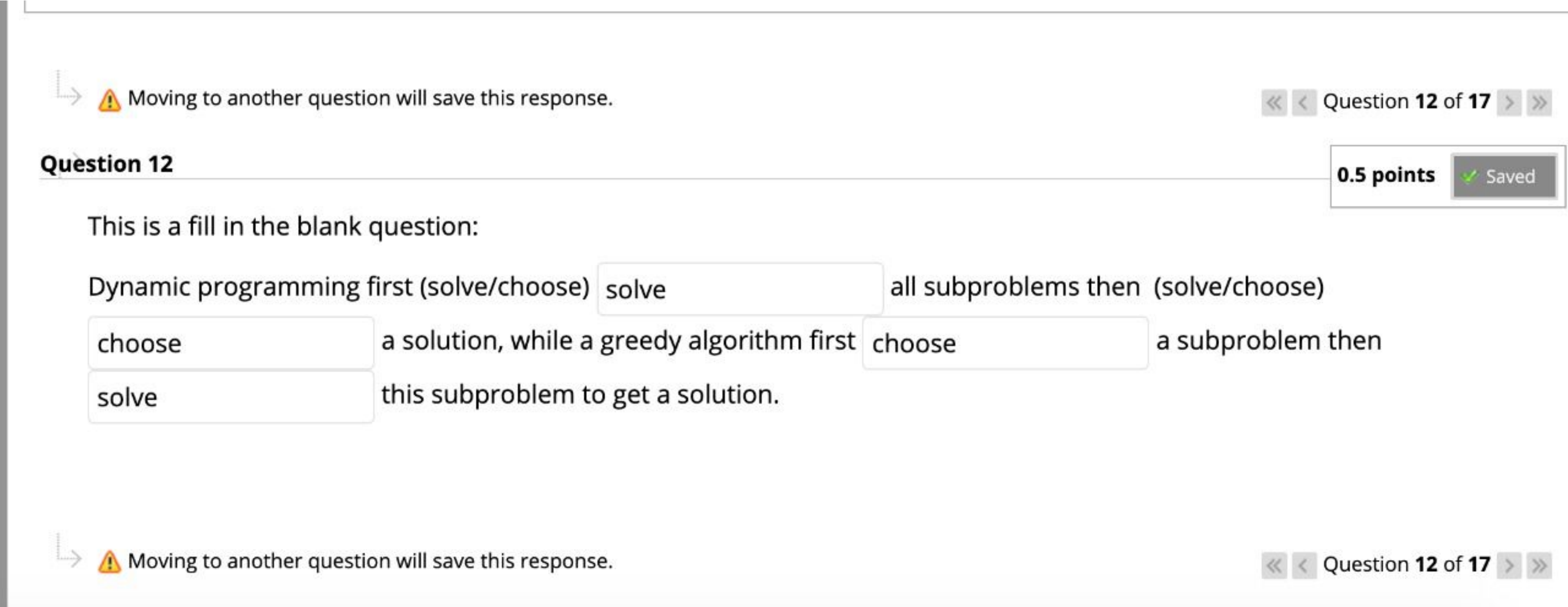
d) After the algorithm terminates, the resultant code tree will have a root node with the value 220 , the code for the letter B will be 00 , and for letter C will be 1101 .

Now answer the following about the Huffman Coding algorithm:

- The design technique of this algorithm is: greedy

- Assuming a binary heap implementation, the running time of this algorithm is: O(nlgn).

- Is its an optimal algorithm (yes/no):B yes .



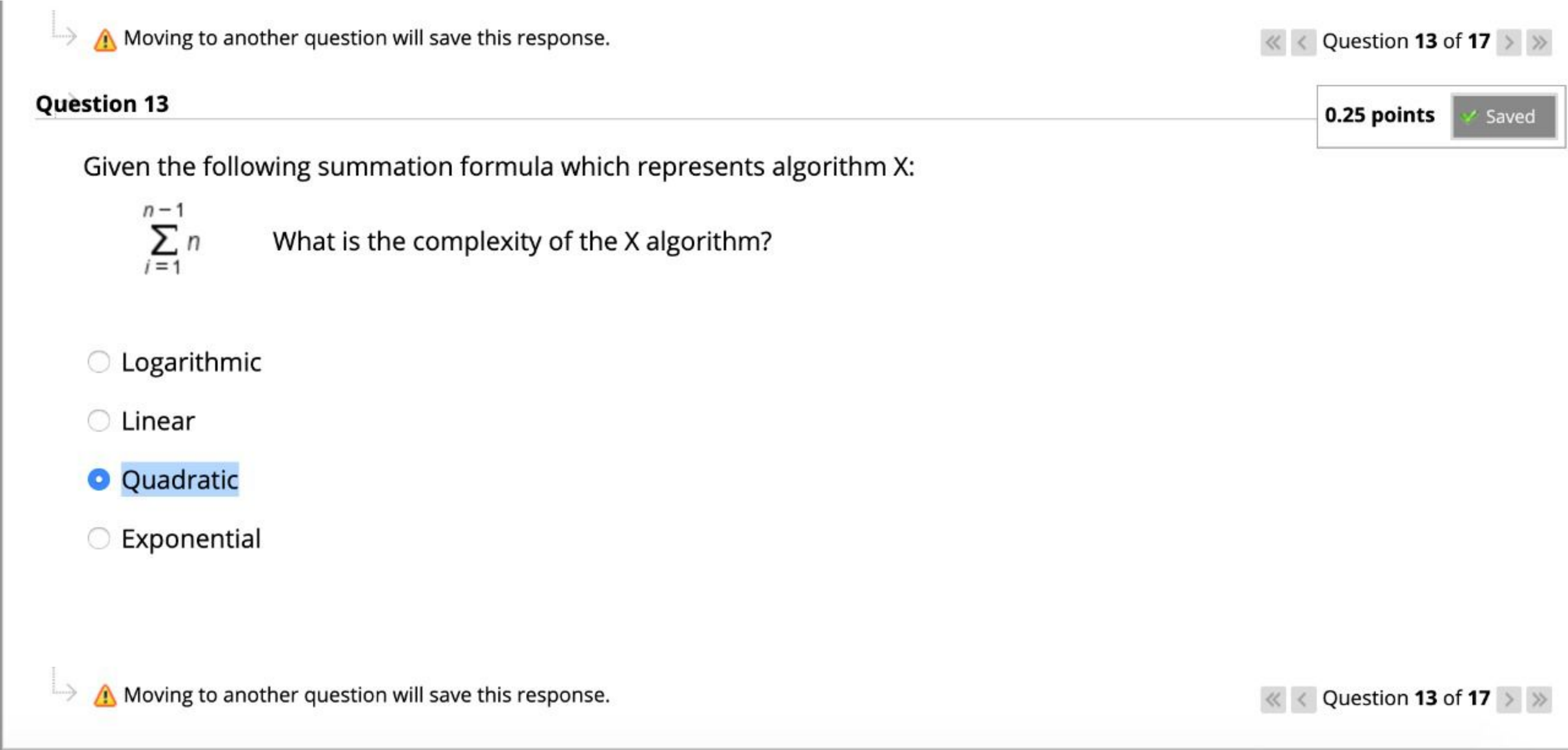
⚠ Moving to another question will save this response.

Question 12

This is a fill in the blank question:

Dynamic programming first (solve/choose) solve all subproblems then (solve/choose) choose a solution, while a greedy algorithm first choose a subproblem then solve this subproblem to get a solution.

⚠ Moving to another question will save this response.



⚠ Moving to another question will save this response.

Question **13** of **17**

Question 13

0.25 points

Given the following summation formula which represents algorithm X:

$$\sum_{i=1}^{n-1} n$$

What is the complexity of the X algorithm?

○ Logarithmic

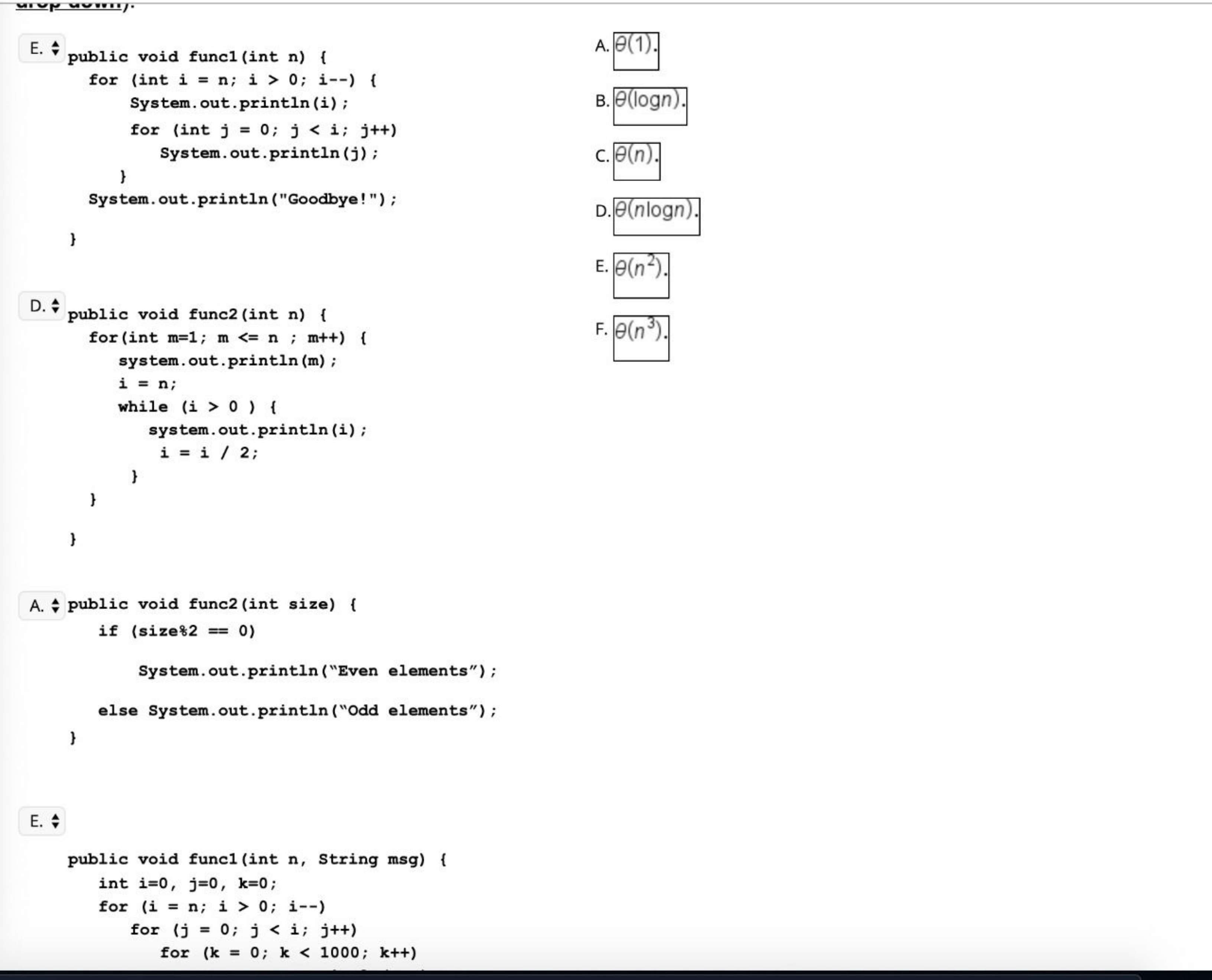
○ Linear

● Quadratic

○ Exponential

⚠ Moving to another question will save this response.

Question **13** of **17**



drop down).

A. $\theta(1)$.

B. $\theta(\log n)$.

C. $\theta(n)$.

D. $\theta(n\log n)$.

E. $\theta(n^2)$.

F. $\theta(n^3)$.

E. ⬍

```
public void func1(int n) {
  for (int i = n; i > 0; i--) {
      System.out.println(i);
      for (int j = 0; j < i; j++)
         System.out.println(j);
     }
  System.out.println("Goodbye!");

}
```

D. ⬍

```
public void func2(int n) {
  for(int m=1; m <= n ; m++) {
     system.out.println(m);
     i = n;
     while (i > 0 ) {
        system.out.println(i);
         i = i / 2;
      }
  }

}
```

A. ⬍

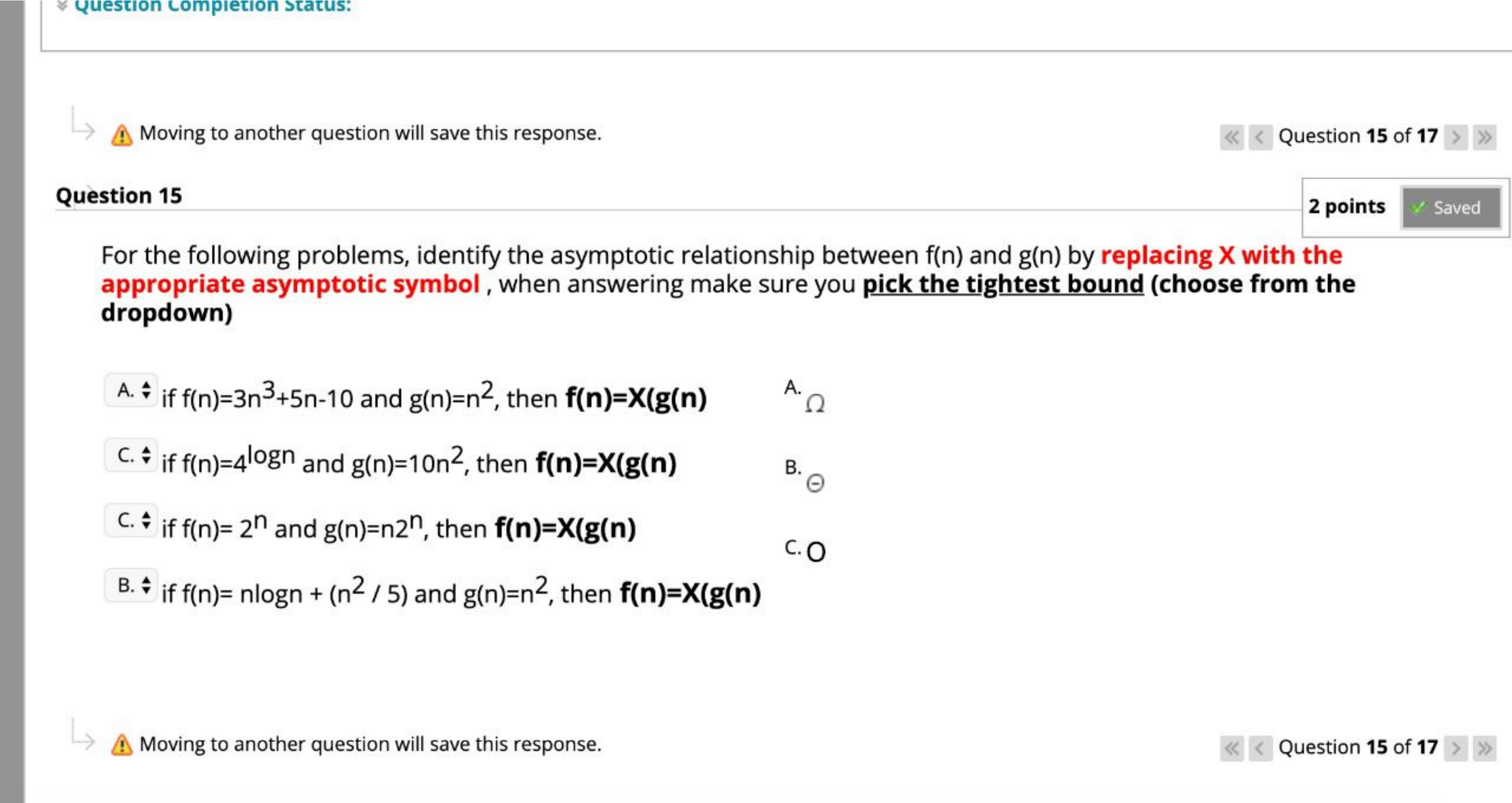
```
public void func2(int size) {
   if (size%2 == 0)

      System.out.println("Even elements");

   else System.out.println("Odd elements");
}
```

E. ⬍

```
public void func1(int n, String msg) {
   int i=0, j=0, k=0;
   for (i = n; i > 0; i--)
      for (j = 0; j < i; j++)
         for (k = 0; k < 1000; k++)
```



Question Completion Status:

⚠ Moving to another question will save this response.

« ‹ Question **15** of **17** › »

Question 15

2 points ✓ Saved

For the following problems, identify the asymptotic relationship between f(n) and g(n) by **replacing X with the appropriate asymptotic symbol** , when answering make sure you **pick the tightest bound** **(choose from the dropdown)**

A. ⬍ if $f(n)=3n^3+5n-10$ and $g(n)=n^2$, then **f(n)=X(g(n)**

C. ⬍ if $f(n)=4^{logn}$ and $g(n)=10n^2$, then **f(n)=X(g(n)**

C. ⬍ if $f(n)= 2^n$ and $g(n)=n2^n$, then **f(n)=X(g(n)**

B. ⬍ if $f(n)= nlogn + (n^2 / 5)$ and $g(n)=n^2$, then **f(n)=X(g(n)**

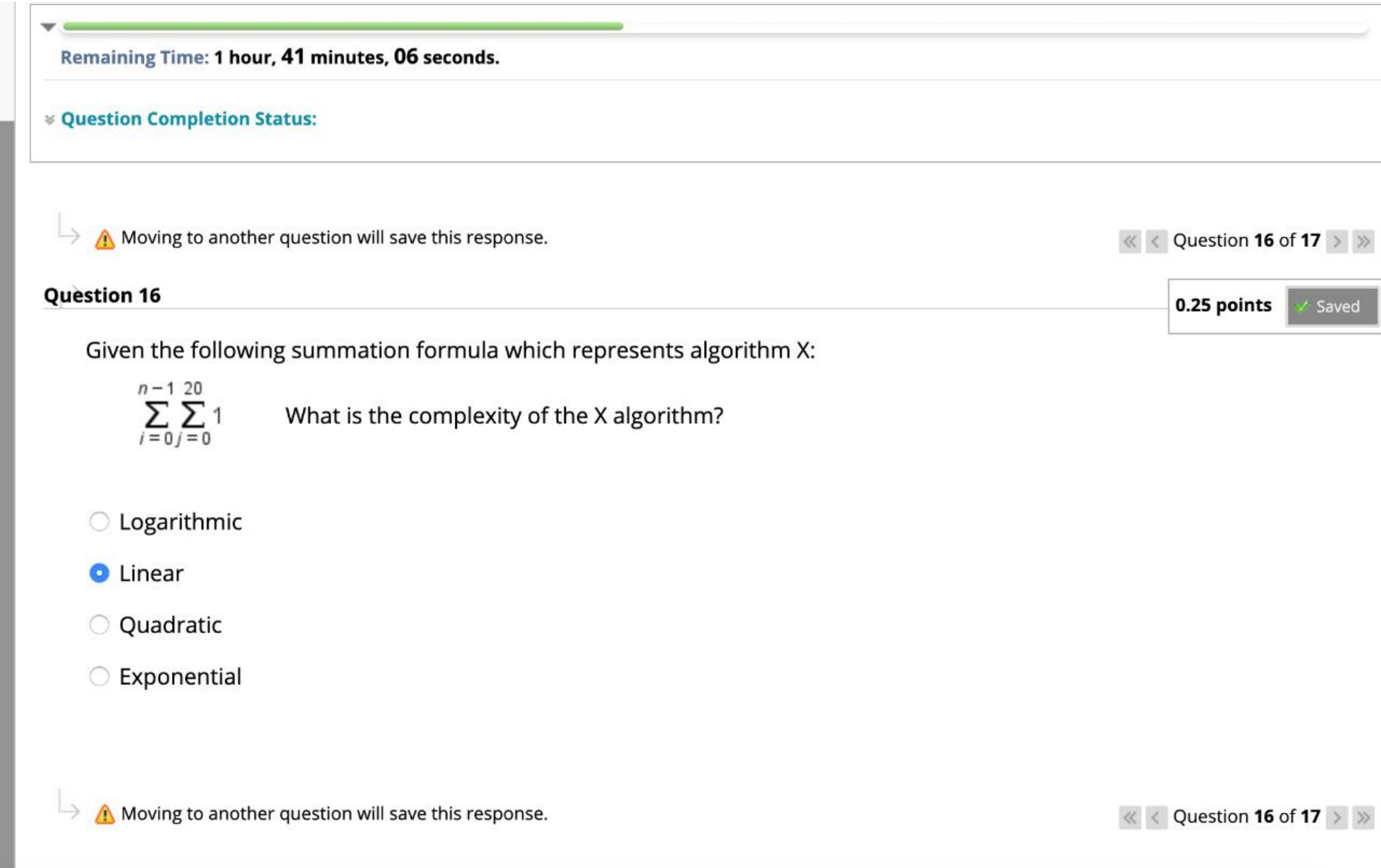
A. Ω

B. Θ

C. O

⚠ Moving to another question will save this response.

« ‹ Question **15** of **17** › »



Remaining Time: **1 hour, 41 minutes, 06 seconds.**

Question Completion Status:

Moving to another question will save this response.

Question **16** of **17**

Question 16

0.25 points ✓ Saved

Given the following summation formula which represents algorithm X:

$$\sum_{i=0}^{n-1}\sum_{j=0}^{20} 1$$

What is the complexity of the X algorithm?

○ Logarithmic

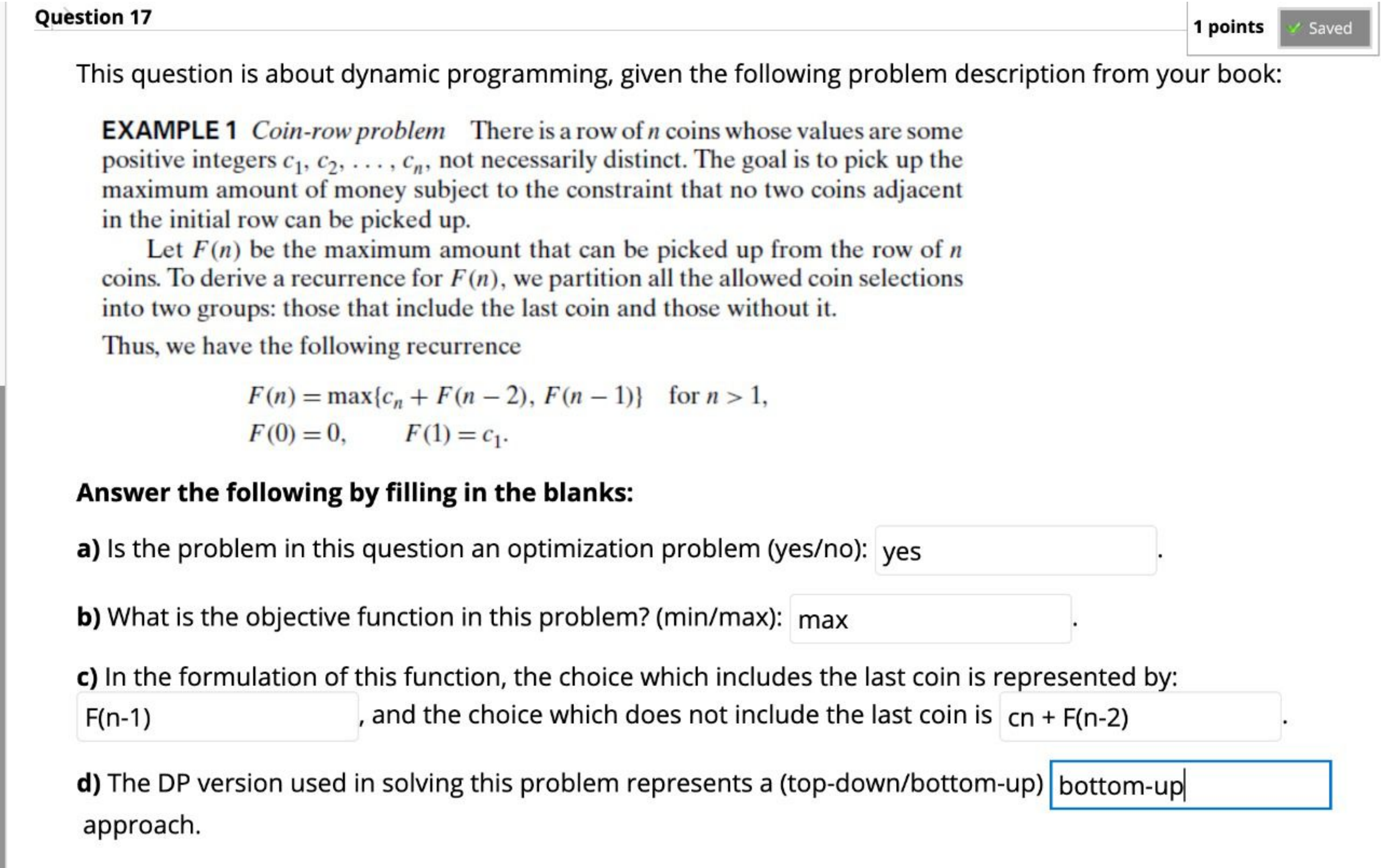
● Linear

○ Quadratic

○ Exponential

Moving to another question will save this response.

Question **16** of **17**



Question 17

1 points ✓ Saved

This question is about dynamic programming, given the following problem description from your book:

EXAMPLE 1 *Coin-row problem* There is a row of n coins whose values are some positive integers $c_1, c_2, \ldots, c_n$, not necessarily distinct. The goal is to pick up the maximum amount of money subject to the constraint that no two coins adjacent in the initial row can be picked up.

Let $F(n)$ be the maximum amount that can be picked up from the row of n coins. To derive a recurrence for $F(n)$, we partition all the allowed coin selections into two groups: those that include the last coin and those without it.

Thus, we have the following recurrence

$$F(n) = \max\{c_n + F(n-2),\ F(n-1)\} \quad \text{for } n > 1,$$
$$F(0) = 0, \qquad F(1) = c_1.$$

Answer the following by filling in the blanks:

a) Is the problem in this question an optimization problem (yes/no): yes .

b) What is the objective function in this problem? (min/max): max .

c) In the formulation of this function, the choice which includes the last coin is represented by: F(n-1) , and the choice which does not include the last coin is cn + F(n-2) .

d) The DP version used in solving this problem represents a (top-down/bottom-up) bottom-up approach.